

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

**ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ
ТЕХНИКИ**

ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине
‘Распределённые системы хранения данных’

Вариант №39645

Выполнил:
Студент группы Р33312
Соболев Иван
Александрович
Преподаватель:
Осипов Святослав
Владимирович

Задание:

Этап 1. Резервное копирование

- Настроить резервное копирование с основного узла на резервный следующим образом:

Периодические полные копии + непрерывное архивирование.
Включить для СУБД режим архивирования WAL; настроить копирование WAL (scp) на резервный узел; настроить полное резервное копирование (pg_basebackup) по расписанию (cron) раз в неделю. Созданные полные копии должны сразу копироваться (scp) на резервный хост. Срок хранения копий на основной системе - 1 неделя, на резервной - 4 недели. По истечении срока хранения, старые архивы и неактуальные WAL должны автоматически уничтожаться.

- Подсчитать, каков будет объем резервных копий спустя месяц работы системы, исходя из следующих условий:
 - Средний объем новых данных в БД за сутки: **650МБ**.
 - Средний объем измененных данных за сутки: **150МБ**.
- Проанализировать результаты.

Этап 2. Потеря основного узла

Этот сценарий подразумевает полную недоступность основного узла. Необходимо восстановить работу СУБД на РЕЗЕРВНОМ узле, продемонстрировать успешный запуск СУБД и доступность данных.

Этап 3. Повреждение файлов БД

Этот сценарий подразумевает потерю данных (например, в результате сбоя диска или файловой системы) при сохранении доступности основного узла. Необходимо выполнить полное восстановление данных из резервной копии и перезапустить СУБД на ОСНОВНОМ узле.

Ход работы:

- Симулировать сбой:
 - удалить с диска директорию любого табличного пространства со всем содержимым.
- Проверить работу СУБД, доступность данных, перезапустить СУБД, проанализировать результаты.
- Выполнить восстановление данных из резервной копии, учитывая следующее условие:
 - исходное расположение дополнительных табличных пространств недоступно - разместить в другой директории и скорректировать конфигурацию.
- Запустить СУБД, проверить работу и доступность данных, проанализировать результаты.

Этап 4. Логическое повреждение данных

Этот сценарий подразумевает частичную потерю данных (в результате нежелательной или ошибочной операции) при сохранении доступности основного узла. Необходимо выполнить восстановление данных на ОСНОВНОМ узле следующим способом:

- Генерация файла на резервном узле с помощью `pg_dump` и последующее применение файла на основном узле.

Ход работы:

- В каждую таблицу базы добавить 2-3 новые строки, зафиксировать результат.
- Зафиксировать время и симулировать ошибку:
 - перезаписать строки любой таблицы “мусором” (`INSERT`, `UPDATE`)
- Продемонстрировать результат.
- Выполнить восстановление данных указанным способом.
- Продемонстрировать и проанализировать результат.

Выполнение:

1. Резервное копирование

Подключение к узлам:

```
ssh -J s336760@helios.cs.ifmo.ru:2222 postgres6@pg157
```

```
ssh -J s336760@helios.cs.ifmo.ru:2222 postgres1@pg158
```

Создаем пользователя для будущих реплик:

```
create role replica replication login password '1234';
```

postgresql.conf

```
wal_level = replica                                # minimal, replica, or logical

archive_mode = on                                  # enables archiving; off, on, or always
                                                    # (change requires restart)
archive_command = 'scp %p postgres1@pg158:/'      # command to use to arc
```

Основной узел:

```
#!/usr/local/bin/bash
```

```
BACKUPS_DIR="/var/db/postgres6/backups"  
TABLE_SPACE="/var/db/postgres6/zzo32"  
CURRENT_DATE=$(date "+%Y-%m-%d-%H:%M:%S")  
BACKUP_DIR=$BACKUPS_DIR/BACKUP_${CURRENT_DATE}
```

```
DATE=$(date "+%Y%m%d%H%M%S")  
BACKUP_NAME="backup_${DATE}"
```

```
NEW_TABLE_SPACE="/var/db/postgres6/tables_backups/${BACKUP_NAME}"
```

```
pg_basebackup -h 127.0.0.1 -p 9455 -U postgres6 -D $BACKUP_DIR -T  
"${TABLE_SPACE}"="${NEW_TABLE_SPACE}"
```

```
scp -r $BACKUP_DIR postgres1@pg158:~/backups/${BACKUP_NAME}/  
scp -r $NEW_TABLE_SPACE  
postgres1@pg158:~/tables_backups/${BACKUP_NAME}/
```

```
SECONDS_IN_WEEK=$(( 7 * 24 * 3600 ))
```

```
current_time=$(date +%s)  
for backup_dir in "$BACKUP_DIR"/*; do  
    file_modified_time=$(stat -f %m "$backup_dir")  
    time_diff=$((current_time - file_modified_time))  
    if [ "$time_diff" -gt "$SECONDS_IN_WEEK" ]; then  
        rm -rf "$backup_dir"  
        echo "file(dir) deleted $backup_dir"  
    fi  
done
```

```
for tables_dir in "$TABLE_SPACE"/*; do  
    file_modified_time=$(stat -f %m "$tables_dir")  
    time_diff=$((current_time - file_modified_time))  
    if [ "$time_diff" -gt "$SECONDS_IN_MONTH" ]; then  
        rm -rf "$tables_dir"  
        echo "file(dir) deleted $tables_dir"  
    fi  
done
```

```
ssh postgres1@pg158 "bash ~/remove_script.sh"
```

Резервный узел:

```
#!/usr/local/bin/bash
BACKUPS_DIR=~/.backups/

SECONDS_IN_MONTH=$(( 28 * 24 * 3600 ))
current_time=$(date +%s)
for backup_dir in "$BACKUPS_DIR"/*; do
    file_modified_time=$(stat -f %m "$backup_dir")
    time_diff=$((current_time - file_modified_time))
    if [ "$time_diff" -gt "$SECONDS_IN_MONTH" ]; then
        rm -rf "$backup_dir"
        echo "file(dir) deleted $backup_dir"
    fi
done
```

На основном узле создадим cron-файл через команду (crontab -e), в котором опишем правило для запуска нашего скрипта раз в неделю по воскресеньям в 00:00:00. Проверим список запланированных задач:

```
[postgres6@pg157 ~]$ crontab -l
0 0 * * sun /var/db/postgres6/create_script.sh >> logfile
```

Проверим работу скрипта:

```
[postgres6@pg157 ~/zso32]$ bash ~/create_script.sh
Password for postgres1@pg158.cs.ifmo.ru:
postgresql.conf 100% 28KB 32.6MB/s 00:00
replorigin_checkpoint 100% 8 42.1KB/s 00:00
pg_ident.conf 100% 1636 7.6MB/s 00:00
logfile 100% 1878 7.4MB/s 00:00
00000001000000000000000012 100% 16MB 147.7MB/s 00:00
postgresql.auto.conf 100% 88 415.7KB/s 00:00
current_logfiles 100% 44 155.1KB/s 00:00
2608_fsm 100% 24KB 38.0MB/s 00:00
2579 100% 16KB 29.2MB/s 00:00
2674_fsm 100% 24KB 42.3MB/s 00:00
4172 100% 8192 17.1MB/s 00:00
0558_fsm 100% 24KB 42.3MB/s 00:00
```

Резервная копия создалась:

```
[postgres1@pg158 ~/.backups]$ ls
backup_20240323142109
```

Расчет объема:

Размер одного бэкапа (изначально):

```
[postgres1@pg158 ~/.backups]$ du -sh backup_20240323142109/
9,9M backup_20240323142109/
```

Количество данных для перезаписывания: 650+150 Мб в день. Копии создаются каждое воскресенье, следовательно за месяц будет создано 4 копии. Так как на основном узле копии хранятся 7 дней, а на резервном 28 дней, то на момент прохождения месяца на обоих узлах будет одна копия (потому что в 28 день месяца произойдет очистка обоих узлов)

Посчитаем с помощью арифметической прогрессии размер копии за 30 дней:

$$S = (10 \cdot 2 + 800 \cdot (30 - 1)) / 2 \cdot 30 = 351000 \text{ Мб} = 343 \text{ Гб}$$

Так как wal файлы у нас архивируются, то их объем не будет превышать 1Гб (max_wal_size)

Общий размер = 344 Гб

Потеря основного узла

Воссоздадим файловую структуру кластера для восстановления:

```
mkdir -p $HOME/ifg51
```

```
chmod 700 ifg51/
```

```
cp -r ~/backups/backup_20240323160134/* ~/ifg51/
```

Файлы табличного пространства

```
mkdir -p $HOME/zzo32
```

```
chmod 700 zzo32/
```

```
cp -r ~/backups/backup_20240323160134/pg_tblspc/16384/ ~/zzo32/
```

Меняем символические ссылки:

```
ln -s /var/db/postgres1/zzo32/ ~/ifg51/pg_tblspc/16384
```

Укажем в postgresql.conf команду для загрузки wal файлов:

```
restore_command = 'cp /var/db/postgres1/wal_archive/%f %p'
```

Создадим в директории кластера файл, сигнализирующий о восстановлении:

```
touch ~/ifg51/recovery.signal
```

Запускаем резервный кластер: `postgres -D $HOME/ifg51 >~/logfile 2>&1 &`

Проверяем работоспособность: `psql -h localhost -p 9455 -U postgres6 wetbluelove`

```
[postgres1@pg158 ~/backups/backup_20240323160134]$ psql -h localhost -p 9455 -U postgres6 wetbluelove
psql (14.2)
Введите "help", чтобы получить справку.

wetbluelove=# \dt+

```

Схема	Имя	Тип	Владелец	Хранение	Метод доступа	Размер	Описание
public	addresses	таблица	ivan	постоянное	heap	8192 bytes	
public	order_items	таблица	ivan	постоянное	heap	8192 bytes	
public	orders	таблица	ivan	постоянное	heap	8192 bytes	
public	products	таблица	ivan	постоянное	heap	8192 bytes	
public	users	таблица	ivan	постоянное	heap	8192 bytes	

```
(5 строк)
```

Анализ выполнения

Восстановление завершилось успешно. Но для корректного запуска необходимо подкорректировать ссылку

Повреждение файлов БД

Я запустил кластер с одного из бэкапов и удалил табличное пространство.

```
[postgres6@pg157 ~]$ pg_ctl -D /var/db/postgres6/backups/BACKUP_2024-03-23-18\:35\:05/ -l logfile start
ожидание запуска сервера.... готово
сервер запущен
[postgres6@pg157 ~]$ psql -h 127.0.0.1 -p 9455 -U ivan wetbluelove
psql (14.2)
Введите "help", чтобы получить справку.

wetbluelove=> ^Z
```

```
[postgres6@pg157 ~]$ cd backups/BACKUP_2024-03-23-18\:35\:05/
[postgres6@pg157 ~/backups/BACKUP_2024-03-23-18:35:05]$ ls
backup_label.old      logfile                pg_multixact          pg_stat_tmp           pg_xact
backup_manifest       pg_commit_ts          pg_notify              pg_subtrans           postgresql.auto.conf
base                  pg_dynshmem           pg_repslot            pg_tblspc             postgresql.conf
current_logfiles      pg_hba.conf           pg_serial              pg_twophase            postmaster.opts
global                pg_ident.conf         pg_snapshots          PG_VERSION             postmaster.pid
log                   pg_logical            pg_stat                pg_wal
[postgres6@pg157 ~/backups/BACKUP_2024-03-23-18:35:05]$ cd pg_tblspc/
[postgres6@pg157 ~/backups/BACKUP_2024-03-23-18:35:05/pg_tblspc]$ ls
16384
[postgres6@pg157 ~/backups/BACKUP_2024-03-23-18:35:05/pg_tblspc]$ rm -r 16384
```

```
[postgres6@pg157 ~/backups/BACKUP_2024-03-23-18:35:05/pg_tblspc]$ psql -h 127.0.0.1 -p 9455 -U ivan wetbluelove
psql: ошибка: подключиться к серверу "127.0.0.1", порту 9455 не удалось: ВАЖНО: база данных "wetbluelove" не существует
ПОДРОБНОСТИ: Подкаталог базы данных "pg_tblspc/16384/PG_14_202107181/16386" отсутствует.
[postgres6@pg157 ~/backups/BACKUP_2024-03-23-18:35:05/pg_tblspc]$
```

Теперь при попытке подключения возникает ошибка.

Переносим копию на основной узел:

```
[postgres1@pg158 ~/tables_backups/backup_20240323183505/Pg_14_202107181]$ scp -r ~/backups/backup_20240323183505/ postgres6@pg157:~/ifg51/
Password for postgres6@pg157.cs.ifmo.ru:
PG_VERSION
global.stat
db_0.stat
db_16386.stat
current_logfiles
```

Переносим табличные пространства в новую директорию

```
[postgres1@pg158 ~/tables_backups/backup_20240323183505/Pg_14_202107181]$ scp -r ~/tables_backups/backup_20240323183505/Pg_14_202107181/ postgres6@pg157:~/zso32_new/
Password for postgres6@pg157.cs.ifmo.ru:
Password for postgres6@pg157.cs.ifmo.ru:
2838
4164
2602_vm
16615
2832
100% 504KB
100% 8192
100% 8192
100% 8192
100% 8
```

Запускаем поврежденный кластер в режиме восстановления wal-файлов и указываем команду для восстановления

Меняем команду восстановления в файле postgresql.conf для текущего бэкапа:

```
restore_command = 'scp postgres1@158:~/wal_archive/%f %p' # command to use to restore an archived logfile segment
```

Меняем символическую ссылку:

```
ln -s /var/db/postgres6/zzo32_new/ /var/db/postgres6/backups/BACKUP_2024-03-23-18\:35\:05/pg_tblspc/16384
```

Запускаем в режиме восстановления

```
touch ~/backups/BACKUP_2024-03-23-18\:35\:05/recovery.signal
```

```
postgres -D $HOME/backups/BACKUP_2024-03-23-18\:35\:05/ >~/logfile 2>&1 &
```

Результат:

```
[postgres6@pg157 ~/zzo32]$ psql -h 127.0.0.1 -p 9455 -U ivan wetbluelove
psql (14.2)
Введите "help", чтобы получить справку.

wetbluelove=>
```

```
wetbluelove=> \d
```

Список отношений			
Схема	Имя	Тип	Владелец
public	addresses	таблица	ivan
public	addresses_address_id_seq	последовательность	ivan
public	order_items	таблица	ivan
public	order_items_item_id_seq	последовательность	ivan
public	orders	таблица	ivan
public	orders_order_id_seq	последовательность	ivan
public	products	таблица	ivan
public	products_product_id_seq	последовательность	ivan
public	users	таблица	ivan
public	users_user_id_seq	последовательность	ivan

(10 строк)

Анализ

Восстановление завершилось успешно. Но для корректного запуска необходимо подкорректировать ссылку

Логическое повреждение данных

Сначала БД будет наполнена новыми данными, с которыми потом будут производиться неаккуратные действия.


```
[postgres6@pg157 ~]$ psql -h localhost -p 9455 -U ivan wetbluelove
psql (14.2)
Введите "help", чтобы получить справку.

wetbluelove=> select * from products;
 product_id | name      | price
-----+-----+-----
          1 | Ноутбук  | 1000.00
          2 | Смартфон | 500.00
(2 строки)

wetbluelove=> INSERT INTO users (name, email) VALUES ('Иван', 'ivan@example.com');
INSERT INTO users (name, email) VALUES ('Мария', 'maria@example.com');

INSERT INTO products (name, price) VALUES ('Ноутбук', 1000.00);
INSERT INTO products (name, price) VALUES ('Смартфон', 500.00);

INSERT INTO orders (user_id, total_amount) VALUES (1, 1500.00);
INSERT INTO orders (user_id, total_amount) VALUES (2, 800.00);

INSERT INTO order_items (order_id, product_id, quantity) VALUES (1, 1, 2);
INSERT INTO order_items (order_id, product_id, quantity) VALUES (2, 2, 1);

INSERT INTO addresses (user_id, street, city, zip_code) VALUES (1, 'Улица Пушкина, 10', 'Москва', '123456');
INSERT INTO addresses (user_id, street, city, zip_code) VALUES (2, 'Улица Лермонтова, 20', 'Санкт-Петербург', '654321');
```

```
wetbluelove=> select * from products;
 product_id | name      | price
-----+-----+-----
          1 | Ноутбук  | 1000.00
          2 | Смартфон | 500.00
          3 | Ноутбук  | 1000.00
          4 | Смартфон | 500.00
(4 строки)
```

После этого сделаем РК и отправим её на резервный узел:

```
[postgres6@pg157 ~]$ bash create_script.sh
Password for postgres1@pg158.cs.ifmo.ru:
postgres1.conf
```

Далее после запуска на резервном узле появятся следующие данные и та самая пока не «испорченная» БД (данные) на которой будет сделан дамп:

```
[postgres1@pg158 ~/backups/backup_20240323183505]$ psql -h localhost -p 9455 -U ivan wetbluelove
psql (14.2)
Введите "help", чтобы получить справку.

wetbluelove=> select * from products;
 product_id | name      | price
-----+-----+-----
          1 | Ноутбук  | 1000.00
          2 | Смартфон | 500.00
          3 | Ноутбук  | 1000.00
          4 | Смартфон | 500.00
(4 строки)
```

Далее делается сам дамп БД wetbluelove, которая будет испорчена на основном узле:

```
pg_dump -h 127.0.0.1 -p 9455 -U ivan -f ~/dump_file wetbluelove
```

Удалим таблицу с продуктами на основном узле:

```
wetbluelove=> DROP table products CASCADE;
ЗАМЕЧАНИЕ: удаление распространяется на объект ограничение order_items_product_id_fkey в отношении таблица order_items
DROP TABLE
```

Теперь нужно скопировать дамп с резервного узла и применить его на основном:

```
scp dump_file postgres6@pg157:~/
```

Далее восстанавливаем данные из файла в текстовом формате:

```
psql -h 127.0.0.1 -p 9455 -d wetbluelove < ~/dump_file
```

```
[postgres6@pg157 ~]$ psql -h 127.0.0.1 -p 9455 -U ivan wetbluelove
psql (14.2)
```

Введите "help", чтобы получить справку.

```
wetbluelove=> \d
```

Список отношений			
Схема	Имя	Тип	Владелец
public	addresses	таблица	ivan
public	addresses_address_id_seq	последовательность	ivan
public	order_items	таблица	ivan
public	order_items_item_id_seq	последовательность	ivan
public	orders	таблица	ivan
public	orders_order_id_seq	последовательность	ivan
public	products	таблица	ivan
public	products_product_id_seq	последовательность	ivan
public	users	таблица	ivan
public	users_user_id_seq	последовательность	ivan

(10 строк)

```
wetbluelove=> select * from products;
```

product_id	name	price
1	Ноутбук	1000.00
2	Смартфон	500.00
3	Ноутбук	1000.00
4	Смартфон	500.00

(4 строки)

Видно, что таблица опять появилась с теми же данными.

Выводы

Во время выполнения лабораторной работы я изучил способы непрерывного бекапа кластера PostgreSQL, на практике настроил и применил его при различных сбоях.